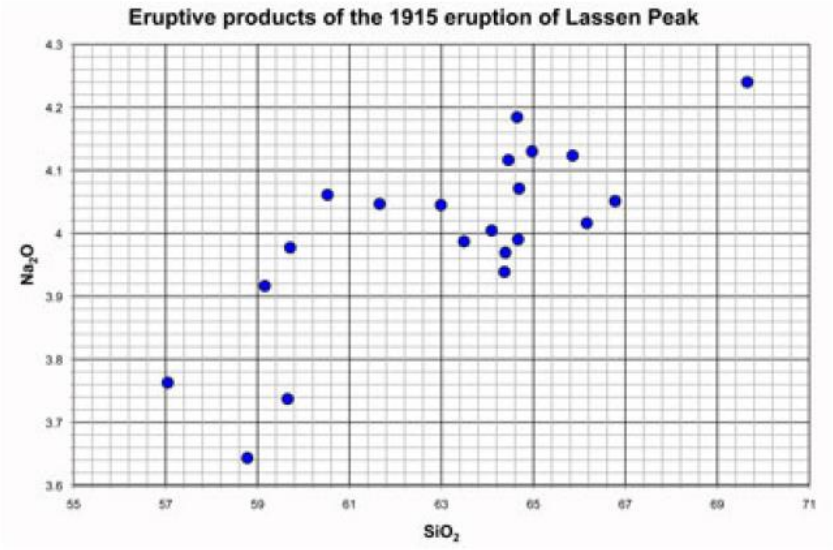
Linear Regression

Background

Suppose we have an independent variable x (time for example). And, we have some other variable y, and we want to ask how the variable y depends on x (maybe y are the profits that year). So, we sample y for different values of x and we get something like this:

Now, we want to determine the relationship between x and y with these samples. Well,

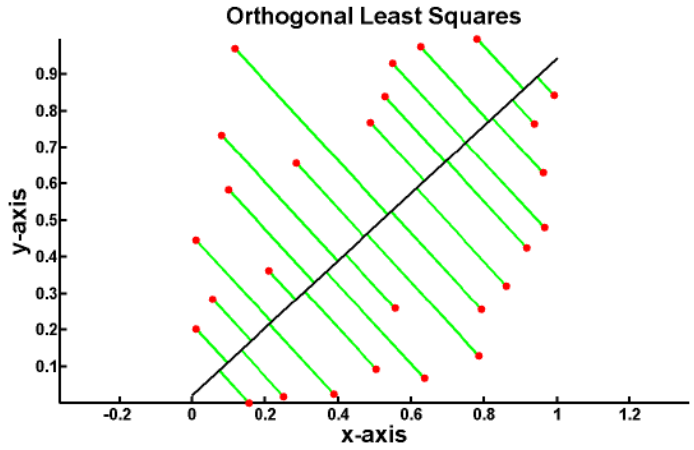


it's hard to say. So, we usually have to make an assumption about the type. The simplest and fairly common case is a *linear* relationship. This leads us to the question - how do we choose a linear function that best describes this data?

This idea is called *linear regression*. In other words, given some data y of our variable x, we wish to find a linear function that accurately captures the data. The linear function we choose isn't unique and based on how we wish to weight the importance of these data points. These linear functions are called *linear predictor functions*. We will focus on one type, which is the least squares approach.

Method of Least Squares

The idea behind least squares is to find the line that passes through all of our data points such that it minimizes the summed distance of all the points to the line. Another way to phrase it is the line which reduces the error of the points as much as possible. If you want to think about it visually, it looks something like this:



where the green lines would each contribute to the 'error'. So, we want to make the sum of all those green lines the least. This is one of the simplest ideas and it makes sense in a lot of practical applications. If our sampled data (y) is measured about the same way, it should have about the same amount of 'noise', so each point is equally important.

In more complex methods, this assumption is relaxed. For example, maybe the measurements get noisier as x increases. In this case, you might wish to use a different error calculation. These ideas are based on what norm you use. We won't be covering this. However, if interested, I suggest taking a statistics class.

Back to this perpendicular line idea - this is called the method of least squares. To begin, we assume the data points obey the equation:

$$y_i = \beta x_i + \alpha + \epsilon_i$$

where β and α the parameters of the line we want to fit and ϵ_i is the noise of that sample. Since ϵ_i is the contributor to the error, the idea is then to minimize the distance between the line $y = \beta x + \alpha$ and our points. This is given by:

$$\min_{(\beta,\alpha)} \sum_{i=1}^{N} (\beta x_i + \alpha - y_i)^2 = \min_{(\beta,\alpha)} ||\beta \vec{x} + \alpha \vec{1} - \vec{y}||^2 = \min_{(\beta,\alpha)} ||A(\alpha, \beta)^T - y||^2$$

where A in this case is a matrix of the data points x_i as the first column and $1's$ as the second column and y is a vector of the data points y_i. Without going into too much detail, you can show through calculus (take derivatives), that the minimzers (β, α), which is what we're trying to solve for, satisfy:

$$A^T A(\beta, \alpha)^T = A^T y$$

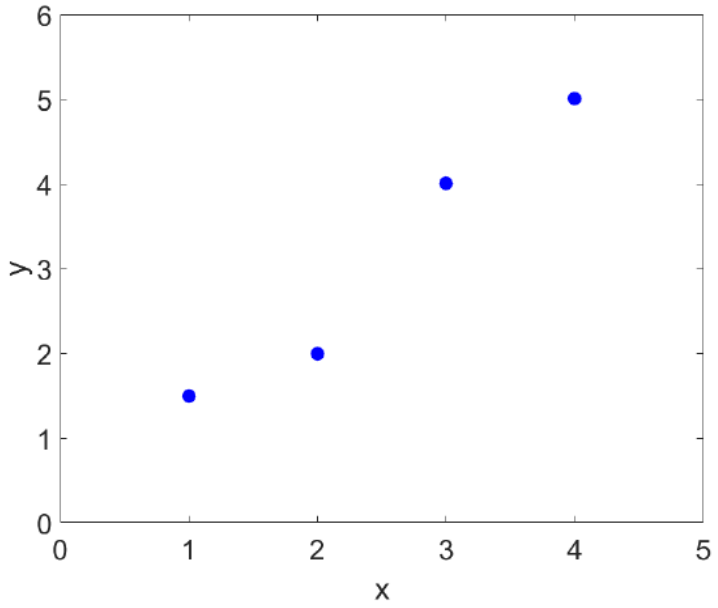
Due to some fairly detailed analysis, we know that A^TA is invertible in this case. So, we can find (β, α) directly by computing:

$$(\beta, \alpha)^T = (A^TA)^{-1}A^Ty$$

In fact, we could extend this to multiple dimensions. The variables (y, z) could be two dependent variables of x. In this case, our line would exist in three dimensions. The analysis is similar, as is the solution. However, for simplicity, we only consider two dimensions. We now discuss an example.

Example of Least Squares

Consider the data points: $(1, 1.5)$, $(2, 2)$ and $(3, 4)$ $(4, 5)$. Plotting looks something like:



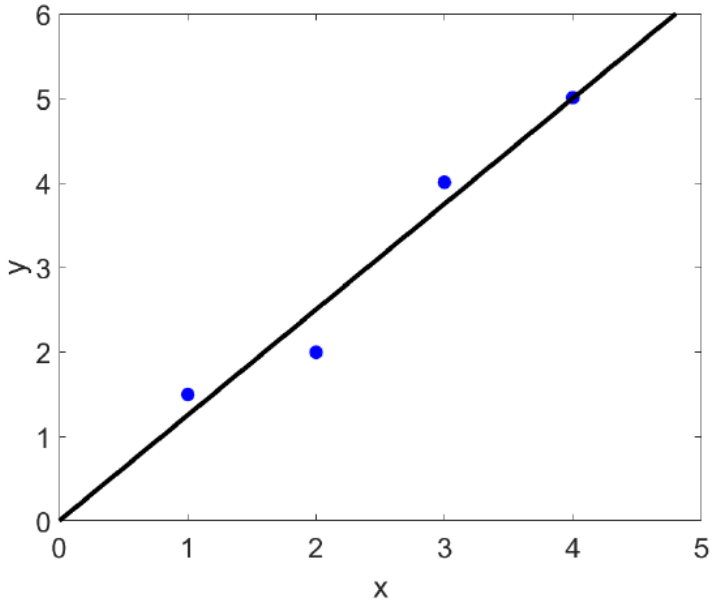
Setting up, our matrix A and vector y should be:

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{bmatrix}, \quad y = \begin{bmatrix} 1.5 \\ 2 \\ 4 \\ 5 \end{bmatrix}$$

Remember, our first column is the x axis data and second column is a vector of 1's for the y-axis. Our vector y is a collection of the y axis data. The element locations for x and y should match up. Now, all we need to do to solve for (β, α) is compute the equation above, so:

$$(\beta, \alpha)^T = (A^TA)^{-1}A^Ty = (5/4, 0)^T$$

Remember, our line satisfies $y = \beta x + \alpha$, so we now know our line is: $y = (5/4)x$. We also plot this and compare against our points:



And that's it. The black line is the best fit line (under the 2-norm) to our four data points. From this, we could also make predictions, such as what the value of y should be at $x = 6$, for example. You could also include more data points and see how that effects the line.

Least Squares and MATLAB

Right, so let's discuss how to do the previous section's example in MATLAB. Then, we can discuss how to do so in general, where we include external data sets which we import. So, for the example, we need to store the x values and y values. Let's do so:

```
>> x = [1;2;3;4]; y = [1.5;2;4;5];
```

Remember, the element locations need to match! Index i for x and y should give the pair of points (x_i, y_i). Now, we need to build the matrix A. This matrix is going to be a column containing our x data and a column of 1's. So,

```
>> A = [x,ones(4,1)];
```

Lastly, we want β and α, so:

```
>> b = inv(A'*A)*A'*y;
```

Now our parameters (β, α) are stored in b. Remember, the first element of b is β and the second element is α. So, to make the line, let's make some grid points:

```
>> xline = 0:.01:1;
```

and then compute the corresponding y value of our line using b:

```
>> yline = b(1)*xline + b(2);
```

From there, we're done. All that's left is plotting, which is left as an exercise to the student.

Now, one thing that may need to be done is importing data. We've done this throughout the semester, but let's review. Suppose we have some data contained in the file sampledata.txt. We can import this with the function `csvread()`, like so:

```
>> DATAVAR = csvread('sampledata.txt');
```

Now, you'll need to check how this is stored and how to import it. For example, row 1 might be the x data and row 2 the y data. But, it could also be column 1 as the x data and column 2 as the y data. It's important to read the size of your variable to determine which is which. We assume the columns are the x and y data. So, we can extract them:

```
>> x = DATAVAR(:,1); y = DATAVAR(:,2);
```

Then, from there, we can repeat what we've done previously. Your assignment is based on this procedure.

Some Extensions

A nice fact about linear regression is that it can sometimes work on nonlinear data sets. For example, suppose you're looking at population growth models, i.e. cell division in time. This is exponential growth and something you've seen in your Calculus sequence. Remember, these types of equations looked like:

$$y = Ce^{rx}$$

where C depended on the initial condition and r was the growth rate. If we wanted to estimate this parameters by taking measurements, we can actually use linear regression! So, suppose we make measurements with some noise:

$$y_i = (C\epsilon_i)e^{rx_i}$$

Notice we've actually changed the noise - it's multiplicative instead of additive. This is an important consideration in doing linear regression properly, but we will gloss over the details for now. Now, we will do a simple trick. Define measurements $z_i := \log(y_i)$ and apply natural logarithms to both sides of the equation. Then, this reduces to:

$$z_i = rx_i + \log(C) + \log(\epsilon_i)$$

Notice this is what we've had before in the linear case, where now $\beta = r$ and $\alpha = \log(C)$. Again, our noise has changed a bit. So, the linear predictor function could be different. However, once we have our linear fit, we can convert it back and get a fit to an exponential function. This is a nice result - linear regression can extend to nonlinear problems.